Foundations of Software Testing and Validation by Coursera

**Exercise: Designing a basic test plan for a software scenario**

## **Software scenario**

Imagine a simple web-based calculator app that can perform basic arithmetic operations such as addition, subtraction, multiplication and division. The calculator has a user interface with buttons for digits 0-9, decimal point and arithmetic operators (+, -, ×, ÷). There is also a display field for input and output, a "Clear" button to reset the display and an "Equals" button to perform the calculation.

### **1. Test planning**

a. Objectives:

* To verify that the calculator app correctly performs basic arithmetic operations: addition, subtraction, multiplication, and division.
* To ensure the app’s user interface functions as expected, including input handling and the functionality of buttons.

Verify that the calculator application functions correctly and has a user- friendly interface.

* *Verify that addition(+) functionality is working as expected.*
* *Verify that subtraction(-) functionality is working as expected.*
* *Verify that multiplication(\*) functionality is working as expected.*
* *Verify that division(/) functionality is working as expected.*
* *Verify that the “Equals” button is working as expected.*
* *Verify that the “Clear” button is working as expected.*
* *Verify that the “Input” is working as expected.*
* *Verify that the “Output” is working as expected.*

b.  **Scope** The scope of testing includes:

* **Arithmetic Operations**: Addition, subtraction, multiplication, and division.
* **User Interface**: Functionality of digit buttons, arithmetic operators, decimal point, "Clear" button, and "Equals" button.
* **Edge Cases**: Division by zero, handling of decimal numbers, and multiple sequential operations.
* **Error Handling**: Correct display or message when invalid operations are performed (e.g., division by zero).

c. Approach:

* **Manual Testing**: Given the simplicity of the application, manual testing will be performed. This will include both positive and negative test cases.
* **Test Types**:
  + Positive test cases will verify correct operations and expected results.
  + Negative test cases will check how the application handles erroneous or edge inputs.

### **2. Test case design**

a. Positive test cases:

1. Test valid *input combinations for addition, subtraction, multiplication and division*.
2. *Test decimal calculations.*
3. *Test the "Clear" button functionality.*

b. Negative test cases:

1. *Test invalid input combinations, such as multiple decimal points or consecutive operators.*
2. *Test division by zero scenario and proper error message display.*
3. *Test input limit for the display field.*

### **3. Test execution**

a. ***Manual Execution:***

* ***Execution****: Perform the test cases step-by-step as outlined, entering values and operators into the calculator, then pressing Equals and Clear where applicable.*
* ***Validation****: Compare the displayed results with expected results for each test case.*

*b.* ***Defect Documentation:***

* ***Recording****: Record any discrepancies between the actual and expected outputs as defects, along with relevant details such as input values, expected output, actual output and steps to reproduce the issue.*
* ***Attachments:*** *Document any discrepancies between actual results and expected results, including screenshots if necessary.*

### **4. Test evaluation**

a. ***Defect Analysis:***

* ***Review****: Analyze each defect based on its severity and impact on functionality.*
* ***Severity****: Classify defects into categories such as critical (app crashes or incorrect results), major (significant functional issues), or minor (cosmetic or non-critical issues).*
* ***Impact****: Assess how defects affect the user experience and overall functionality.*

b. ***Decision Making:***

* ***Fix Before Release****: Critical and major defects should be addressed before release to ensure the app is reliable and functional.*
* ***Future Update****: Minor defects or those that don’t significantly impact functionality might be deferred for fixing in future updates or patches.*